Quantitative Finance

ISSN: 1469-7688 (Print) 1469-7696 (Online) Journal homepage: https://www.tandfonline.com/loi/rquf20

Bayesian regularized artificial neural networks for the estimation of the probability of default

Eduard Sariev & Guido Germano

To cite this article: Eduard Sariev & Guido Germano (2019): Bayesian regularized artificial neural networks for the estimation of the probability of default, Quantitative Finance, DOI: 10.1080/14697688.2019.1633014

To link to this article: https://doi.org/10.1080/14697688.2019.1633014

Published online: 31 Oct 2019.

Quantitative Finance, 2019
https://doi.org/10.1080/14697688.2019.1633014

Bayesian regularized artificial neural networks for the estimation of the probability of default

EDUARD SARIEV*† and GUIDO GERMANO†‡

†Financial Computing and Analytics Group, Department of Computer Science, University College London, Gower Street, London, WC1E 6BT, UK
‡Systemic Risk Centre, London School of Economics and Political Science, Houghton Street, London, WC2A 2AE, UK

(*Received 1 November 2018; accepted 29 April 2019; published online 31 October 2019*)

Artificial neural networks (ANNs) have been extensively used for classification problems in many areas such as gene, text and image recognition. Although ANNs are popular also to estimate the probability of default in credit risk, they have drawbacks; a major one is their tendency to overfit the data. Here we propose an improved Bayesian regularization approach to train ANNs and compare it to the classical regularization that relies on the back-propagation algorithm for training feed-forward networks. We investigate different network architectures and test the classification accuracy on three data sets. Profitability, leverage and liquidity emerge as important financial default driver categories.

Keywords: Artificial neural networks; Bayesian regularization; Credit risk; Probability of default

JEL Classification: C11, C13

1. Introduction

Credit scoring became popular in the USA during the 1950s. The boosting economy in the next two decades required the need for accessible credit and it was during this period when the methods used for automated credit scoring became more advanced (Tufféry 2011). However, the origin of credit scoring goes back to the early 1940s in the USA, when it was initially applied to differentiate between good and bad customers (Durand 1941).

Among the many options offered and investigated in the literature for credit scoring, artificial neural networks (ANNs) are a flexible and rich concept to solve not only classification problems but also to offer solutions to clustering, time series and function approximation problems (Bell 2015). The flexibility of ANNs inspired researchers to investigate their applicability to classification tasks. Recently, an extensive research has been conducted to utilize and apply ANNs for corporate credit scoring given the large amount of financial data collected. Heaton *et al.* (2016) and Pérez-Martín *et al.* (2018) advocated the extensive use of ANNs with many layers, the so-called deep learning approach. Furthermore, Bonini and Caivano (2018) showed that artificial intelligence methods including ANNs outperform traditional statistical methods. Nonetheless, the performance advantages of ANNs are questioned by Kalayci *et al.* (2018) and by Addo *et al.* (2018), who show that ANNs underperform when compared to respectively logistic regression and decision trees.

Here we focus on the overifitting issue of ANNs. Several recent studies have been devoted to this problem. Zhang *et al.* (2018) investigated various ways of detecting overfitting in an ANN and advocated splitting the data into training and validation as a main way of dealing with overfitting. Using a genetic algorithm, Nicolae-Eugen (2016) prevented overfitting in an ANN by encoding the weights of the ANN into binary chromosomes and applying high-probability mutation in the genetic algorithm. A different approach to avoid overfitting was proposed by Vincent *et al.* (2010). They applied a drop-out strategy combined with a stacked denoising autoencoder to reduce overfitting. They found that this strategy outperforms a single drop-out strategy and is computationally more efficient. One of the reasons for overfitting is the noise in the training data. In this context, Hindi and Al-Akhras (2011) recommended smoothing the decision boundaries by eliminating border instances from the training set before training an ANN. This is achieved by using a variety of instance reduction techniques.

*Corresponding author. Email: eduard.sariev.11@ucl.ac.uk

In contrast to these studies on overfitting in ANNs, we take a Bayesian approach to solve the issue. Bayesian estimation in ANNs has become applicable only since the advancement of computational power has allowed its use. Initially, Bayesian learning in ANNs was used to offer a solution for creating an optimal network architecture. For example, Neal (1992) explored the difficulties related to the selection of the prior knowledge as well as the problems associated with the computation of the posterior distribution. Neal (1996) studied the effect of using different priors for the estimation of the network weights. Rasmussen (1996) investigated how to estimate the weights of a network using dynamic simulation. Furthermore, Lampinen and Vehtari (2001) applied an ANN with Bayesian learning to regression and classification. Titterington (2004) reviewed the various approaches taken to determine the network architecture, involving the use of Gaussian approximations and of non-Gaussian but deterministic approximations called variational approximations.

The Bayesian estimation of an ANN for credit scoring implies that the optimal architecture of the neural network is important to the performance because the architecture greatly impacts the estimation efficiency of the network (Heaton *et al.* 2016). However, in this study, we focus on the Bayesian regularization of the network in order to avoid overfitting. We compare our approach to the classical regularization approach examined in Ashiquzzaman *et al.* (2017). For that reason, we report our results as an average performance over a range of different network architectures, i.e. combinations of layers and neurons. Overfitting is one of the main challenges faced by statisticians today. The volume and the complexity of the data increase every year, which requires special attention to not overfit the classification algorithm. In this work, we use a combination of different network architectures combined with early stopping (ES) and regularization to tackle the problem of overfitting. We define ES as the process where we monitor the test error in n consecutive runs, while training the network. If the test error increases n times then the training of the network is terminated. We used $n = 6$, which is a typical choice for most classification problems.

Drawing on the studies devoted to Bayesian learning, in this paper, we improve an approach recommended by MacKay (1992) to estimate the regularization parameters. MacKay (1992) proposed a Gauss–Newton approximation to the posterior distribution of the regularization parameters. In this Gauss–Newton approximation, an objective function with parameters α and β is maximized. MacKay (1992) proposed an iterative solution for α and β by applying the Levenberg–Marquardt algorithm. We on the other hand apply a MCMC scheme to estimate the regularization parameters. In our approach, α and β are considered random variables and are based on the mean of a posterior distribution. Finally, we compare the improved Bayesian regularization approach to classical regularization and to Bayesian regularization based on the Gauss–Newton approximation. With respect to the above discussed articles on ANNs, we contribute to the literature first by proposing an update to the estimation of the regularization parameters and secondly by exploring classical and Bayesian regularization in the estimation of a network with different architectures.

The rest of the article is organized as follows. Section 2 presents the theoretical formulation of an ANN in a classical and in a Bayesian framework. Section 3 presents the results from the regularized networks. Section 4 discusses the policy implications of the selected default factors and their business intuition. Finally, Section 5 concludes the paper by summarizing the main findings.

2. Theoretical foundations

In theory, there are several neural network architectures. In practice, most researchers (Demuth *et al.* 2014) focus on three main types: feed-forward, competitive and recurrent networks. While competitive and recurrent networks are definitely an interesting area of research, in this article, we explore the most popular kind of network architecture, the feed-forward network. It is called a feed-forward network because data moves in forward direction only: initially, the data input is processed in the first layer of the network, then it is pushed forward to the next layer until it reaches the final output layer. In a feed-forward network, data is not fed back from a layer to the previous, which instead happens in a recurrent network. A detailed description of a feed-forward network is given in the next section.

2.1. *Feed-forward neural network architecture*

In this section, we briefly introduce the most basic theoretical concepts behind an ANN. A detailed discussion is given in Kim *et al.* (1996). A multilayer ANN can be described as a system with the following elements:

(i) An input data vector $\mathbf{x} \in \mathbb{R}^p$ and a categorical variable $y \in \{0, 1\}$.
(ii) An output $\widehat{y} = P(Y = 1 \mid \mathbf{X} = \mathbf{x})$.
(iii) Layers $k = 1, \ldots, l$ with m units per layer; the layers with $k < l$ are hidden, the layer l is the output layer. Each layer has a bias $b^k \in \mathbb{R}$ and each unit has an activation $h_i^k \in \mathbb{R}$. The units in layer k are connected to those in the previous layer by weights $w_{ij}^k \in \mathbb{R}$, $i, j = 1, \ldots, m$, $k = 1, \ldots, l$.
(iv) A previous layer is defined as layer $k - 1$ with respect to layer k.
(v) The individual inputs $\mathbf{x} \in \mathbb{R}^p$ are each weighted by weights w_{ij}^k. Each neuron i is weighted in each layer k.
(vi) The final output has a bias $b^{l+1} \in \mathbb{R}$ and is connected to the units of the output layer by weights $w_j^{l+1} \in \mathbb{R}$.
(vii) An activation function $s_i^k(.)$ for layer k and unit i. An activation function determines how each node reacts in an ANN and what output each node generates. This output is then used as input for the next node in an iterative procedure until the estimation process converges to a local or global optimum. The most popular choices of activation functions are the logistic sigmoid and the hyperbolic tangent (Farhadi 2017).

Below we present the sequence in which the estimation of the network weights is performed. The first step in the estimation

process of the network weights w is to feed data into the first layer of the network. The unit activations of the first layer are computed from the input data as

$$h_i^1 = s_i^1\left(b^1 + \sum_{j=1}^{p} w_{ij}^1 x_j\right), \tag{1}$$

where $s_i^1(\cdot)$ is an activation function. After receiving the output from the first layer, we can proceed with the estimation of the second layer activation functions. The unit activations of the next layer are computed from those of the previous layer as

$$h_i^{k+1} = s_i^{k+1}\left(b^{k+1} + \sum_{j=1}^{m} w_{ij}^{k+1} h_i^k\right). \tag{2}$$

After reaching the final output by sequentially moving through each hidden layer k, the output probability is estimated as

$$\widehat{y} = b^{l+1} + \sum_{j=1}^{m} w_j^{l+1} h_i^l. \tag{3}$$

In the estimation process described above the activation function $s(\cdot)$ plays a vital role. In our analysis, we apply the logistic function which is the most common non-linear activation function.

$$s(x) = \frac{1}{1+\exp(-x)}. \tag{4}$$

The above estimation process can be described as a learning process where the weights of the network are estimated through learning from the data. In particular, the weights w_{ij}^k for layer k and neuron i in the neural network are estimated sequentially and iteratively. Afterwards, the network performance with weights learned from the training data is monitored on test data.

In order to estimate the network weights a cost function is required. The purpose of the cost function is to serve as an objective to be minimized during the learning process. A typical choice of a cost function is the mean squared error (MSE)

$$E = \frac{1}{N}\sum_{i=1}^{N}(y_i - \widehat{y}_i)^2, \tag{5}$$

where N is the number of observations, i.e. the number of input data vectors and categorical variables. Another popular cost function is the cross entropy (CE)

$$S = -\sum_{i=1}^{N} p_i \log q_i, \tag{6}$$

where p_i and q_i are discrete probabilities.

A common issue in estimating network weights is the overfitting of the network, upon which the network cannot generalize well and subsequently the network performance on new data is poor. When overfitting occurs, the network weights are calculated in a way that maximizes the network performance on the training data but this is achieved through significantly decreasing the performance on the test data. The most common way of solving the overfitting issue that occurs in the estimation process is applying regularization during the estimation (Deng *et al.* 2014). Regularization can be applied to penalize the cost function with the squared sum of the weights so that the generalization performance of the network is maintained. For the MSE cost function, this can be written as

$$E_{\text{reg}} = \gamma\sum_{k=1}^{l}\sum_{i,j=1}^{m}(w_{ij}^k)^2 + (1-\gamma)E = \gamma E_w + (1-\gamma)E, \tag{7}$$

where $\gamma \in (0,1)$ is a regularization constant. Usually the backpropagation algorithm, Dreyfus (1990) is used to estimate the weights. A common optimization algorithm used to make the estimation procedure converge is the gradient descent algorithm.

Although classical regularization as described above works adequately, in this paper, we recommend a Bayesian approach to regularization which we describe in the next section. We advocate that the Bayesian approach to regularization allows for more flexibility by reducing the bias inherent to classical regularization (through the choice of the regularization constant) and therefore leading to a higher performance.

2.2. *A Bayesian approach for feed-forward neural networks*

After we have explained what a feed-forward network is, in this section, we present the theory behind our proposed approach to regularization. The networks are trained using supervised learning, with a training data set of inputs and targets $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)\}$. We choose an interpolating function of the form

$$g(\mathbf{x}) = \sum_{h=1}^{k} w_h \phi_h(\mathbf{x}), \tag{8}$$

where $\phi_h(\mathbf{x})$ are basis functions and w_h are coefficients inferred from the data. We assume that the targets are generated by

$$y_i = g(\mathbf{x}_i) + \epsilon_i, \tag{9}$$

where $g(\mathbf{x}_i)$ is an unknown function and ϵ_i are independent Gaussian random variables with average zero and variance σ^2. The initial objective of the training process is to minimize the sum of squared errors

$$E_D = \sum_{i=1}^{N}\frac{1}{2}(y_i - \widehat{y}_i)^2, \tag{10}$$

where $\widehat{y}_i$ represents the neural network response to observation i.

An extensive work on Bayesian estimation and regularization has been done by MacKay (1992). In summary, the Bayesian regularization requires the Hessian matrix of the objective function. For the MSE cost function and regularization by the sum of squared weights, it follows that the Hessian matrix is a quadratic function and can be approximated using

the Levenberg–Marquardt algorithm (Gill and Murray 1978). The objective function becomes

$$F = \alpha E_W + \beta E_D, \tag{11}$$

where E_W was defined in equation (7), and α and β are objective function parameters.

In the Bayesian framework (Foresee and Hagan 1997), the weights of the network are considered random variables. Given the data, the probability density function of an array $\mathbf{w}$ of network weights is

$$f(\mathbf{w} \mid D, \alpha, \beta, M) = \frac{f(D \mid \mathbf{w}, \beta, M) f(\mathbf{w} \mid \alpha, M)}{f(D \mid \alpha, \beta, M)}, \tag{12}$$

where M is the particular neural network model used; $f(\mathbf{w} \mid \alpha, M)$ is the prior density, which represents our knowledge of the weights before any data is collected; $f(D \mid \mathbf{w}, \beta, M)$ is the likelihood function, which is the probability of the data occurring given the weights; $f(D \mid \alpha, \beta, M)$ is a normalization factor, which guarantees that the total probability is 1.

Under the assumption of Gaussian noise, the probability of the data given the parameters $\mathbf{w}$ is

$$f(D \mid \mathbf{w}, \beta, M) = \frac{\exp(-\beta E_D)}{Z_D(\beta)}, \tag{13}$$

where $Z_D(\beta) = (2\pi/\beta)^{N/2}$, $\beta = 1/\sigma^2$. The density of the prior can be written as

$$f(\mathbf{w} \mid \alpha, M) = \frac{\exp(-\alpha E_W)}{Z_W(\alpha)}, \tag{14}$$

where $Z_W(\alpha) = \int \exp(-\alpha E_W)\, \mathrm{d}\mathbf{w}$. If equations (13) and (14) are substituted into equation (12), we obtain

$$f(\mathbf{w} \mid D, \alpha, \beta, M) = \frac{\exp(-(\beta E_D + \alpha E_W))}{Z_W(\alpha) Z_D(\beta)} = \frac{\exp(-F(\mathbf{w}))}{Z_F(\alpha, \beta)}. \tag{15}$$

where $Z_F(\alpha, \beta) = \int \exp(-F)\, \mathrm{d}\mathbf{w}$. In this Bayesian framework, the optimal weights should maximize the posterior probability.

2.3. *Optimizing the regularization parameters*

After we showed that the weights are a function of the parameters α and β, we optimize the latter using Bayes' theorem,

$$f(\alpha, \beta \mid D, M) = \frac{f(D \mid \alpha, \beta, M) f(\alpha, \beta \mid M)}{f(D \mid M)}. \tag{16}$$

If a uniform prior density $f(\alpha, \beta \mid M)$ is taken for the regularization parameters α and β, then maximizing the posterior is achieved by maximizing the likelihood function $f(D \mid \alpha, \beta, M)$. This likelihood function is the normalization factor in equation (12). Since all probabilities have a Gaussian form, the posterior can be expressed as

$$f(D \mid \alpha, \beta, M) = \frac{f(D \mid \mathbf{w}, \beta, M) f(\mathbf{w} \mid \alpha, M)}{f(\mathbf{w} \mid D, \alpha, \beta, M)} = \frac{Z_F(\alpha, \beta)}{Z_W(\alpha) Z_D(\beta)}. \tag{17}$$

$Z_D(\beta)$ and $Z_W(\alpha)$ are known from equations (13) and (14). $Z_F(\alpha, \beta)$ can be expanded in a Taylor series. Since the objective function has a quadratic shape in the surrounding of the minimum, we can expand $Z_F(\mathbf{w})$ around the minimum point of the posterior density $\mathbf{w}_{\mathrm{MP}}$, where the gradient is zero. We refer to $\mathbf{w}_{\mathrm{MP}}$ as the most probable interpolant and therefore F can be written as

$$F = F(\mathbf{w}_{\mathrm{MP}}) + \tfrac{1}{2}(\mathbf{w} - \mathbf{w}_{\mathrm{MP}})^{\mathsf{T}} \mathbf{H} (\mathbf{w} - \mathbf{w}_{\mathrm{MP}}), \tag{18}$$

where $\nabla^2 E_D = \mathbf{B}$, $\nabla^2 E_W = \mathbf{C}$, $\mathbf{H} = \alpha \mathbf{C} + \beta \mathbf{B}$, $\mathbf{w}_{\mathrm{MP}} = \mathbf{H}^{-1} \mathbf{B} \mathbf{w}_{\mathrm{ML}}$. It follows that Z_F is a Gaussian integral that can be expressed as

$$Z_F = \mathrm{e}^{-F(\mathbf{w}_{\mathrm{MP}})} (2\pi)^{k/2} (\det \mathbf{H})^{-1/2}. \tag{19}$$

Thus we can rewrite the log evidence for α and β as

$$\log f(D \mid \alpha, \beta, A, R) = -\alpha E_W - \beta E_D + \frac{k}{2} \log(2\pi) - \frac{1}{2} \log \det \mathbf{H} - \log Z_W(\alpha) - \log Z_D(\beta). \tag{20}$$

Notice that this expression contains the logarithm of the Occam factor $(2\pi)^{k/2} (\det \mathbf{H})^{-1/2} / Z_W(\alpha)$, which can control the overfitting. Substituting Z_D from equation (13) and Z_W from equation (14),

$$\log f(D \mid \alpha, \beta, A, R) = -\alpha E_W - \beta E_D - \frac{1}{2} \log \det \mathbf{H} + \frac{k}{2} \log \alpha + \frac{N}{2} \log \beta. \tag{21}$$

We differentiate the log evidence with respect to α and β to find the condition that is satisfied at the maximum. Differentiating with respect to α and setting the result equal to zero gives

$$\alpha_{\mathrm{MP}} = \frac{\gamma}{2E_W(\mathbf{w}_{\mathrm{MP}})}; \tag{22}$$

differentiating with respect to β and setting the result equal to zero gives

$$\beta_{\mathrm{MP}} = \frac{N - \gamma}{2E_D(\mathbf{w}_{\mathrm{MP}})}. \tag{23}$$

One step of the calculation is $(\partial/\partial\alpha) \log \det \mathbf{H} = \mathrm{tr}\,(\mathbf{H}^{-1}(\partial \mathbf{H}/\partial\alpha)) = \mathrm{tr}\,(\mathbf{H}^{-1}\mathbf{I}) = \mathrm{tr}\,\mathbf{H}^{-1} = (\mathrm{tr}\,\mathbf{H})^{-1}$, where $\nabla\nabla^{\mathsf{T}} E_W = \mathbf{I}$. Here $\gamma = k - 2\alpha_{\mathrm{MP}}\, \mathrm{tr}\, \mathbf{H}_{\mathrm{MP}}^{-1}$ is the effective number of parameters and k is the total number of parameters in the network. The parameter γ is a measure of how many parameters in the neural network are effectively used in reducing the error function; it can range from zero to k.

Summarizing, the steps required for the Bayesian optimization of the regularization parameters with a quadratic approximation of the Hessian matrix are:

(i) Initialize the parameters α, β and the weights $\mathbf{w}$.
(ii) Take one step of the Levenberg–Marquardt algorithm to minimize the objective function $F(\mathbf{w}) = \alpha E_W + \beta E_D$.
(iii) Compute the effective number of parameters $\gamma = k - 2\alpha \operatorname{tr} \mathbf{H}^{-1}$ using the Gauss–Newton approximation of the Hessian available in the Levenberg–Marquardt training algorithm, $\mathbf{H} = \nabla^2 F(\mathbf{w}) \approx 2\beta \mathbf{J}^{\mathsf{T}}\mathbf{J} + 2\alpha \mathbf{I}_k$, where $\mathbf{J}$ is the Jacobian matrix of the training set errors.
(iv) Compute new estimates of the objective function parameters $\alpha = \gamma / 2E_W(\mathbf{w})$, $\beta = (N - \gamma)/2E_D(\mathbf{w})$.
(v) Iterate steps (ii) through (iv) until convergence.

2.4. *Markov chain Monte Carlo estimation of α and β*

We propose an improvement on the estimation of the regularization parameters developed by MacKay (1992). We advocate applying a Markov chain Monte Carlo (MCMC) scheme to estimate α and β rather than approximating $Z_F(\alpha, \beta)$ and consequently approximating the Hessian matrix to estimate the parameters α, β. Collecting these paramenters into the two-dimensional vector $\mathbf{x}$ and indicating their estimate with $\mathbf{X}$, the MCMC method (Gelfand and Smith 1990) can be described as

(i) Choose the target distribution of $\mathbf{X}$ with density $\pi(\mathbf{x})$.
(ii) Choose the proposal distribution q: for any $\mathbf{x} \in \mathbb{R}^2_+$ we have $q(\mathbf{x} \mid \mathbf{x}) \geq 0$, $\int q(\mathbf{x} \mid \mathbf{x})\, \mathrm{d}\mathbf{x} = 1$.
(iii) Starting with $\mathbf{X}_1$, for $t = 2, 3, \ldots, M$, sample $\mathbf{X}_* \sim q(\cdot \mid \mathbf{X}_{t-1})$.
(iv) Compute $\alpha(\mathbf{X}_* \mid \mathbf{X}_{t-1}) = \min\{1, \pi(\mathbf{X}_*) q(\mathbf{X}_{t-1} \mid \mathbf{X}_*) / \pi(\mathbf{X}_{t-1}) q(\mathbf{X}_* \mid \mathbf{X}_{t-1})\}$.
(v) Sample $U \sim U_{(0,1)}$. If $U < \alpha(\mathbf{X}_* \mid \mathbf{X}_{t-1})$, set $\mathbf{X}_t = \mathbf{X}_*$, otherwise set $\mathbf{X}_t = \mathbf{X}_{t-1}$.

We apply a standard normal prior distribution to the MCMC scheme.

3. Application of neural networks to financial data

As discussed in the literature, neural networks are a powerful concept that can be applied to different problems ranging from function approximation to clustering. There are many articles devoted to the comparison of neural networks to each other or to other algorithms. Specht (1990) investigated probabilistic neural networks; Wang and Peng (2000) explored vector-quantization networks; Stallkamp *et al.* (2012) compared convolutional neural networks with linear discriminant analysis and decision trees. In contrast to the above studies, in our analysis, we focus on the concept of regularization and how it is applied in the context of neural networks. We test the performance of feed-forward networks with and without regularization. Furthermore, we combine the regularization with ES. However, the main focus of the analysis is on the Bayesian approach to regularization for neural networks. In contrast to the classical approach to regularization, in the Bayesian approach, the regularization parameters are inferred from the data. We propose an improvement of the Bayesian estimation over the one suggested by MacKay (1992). Our estimation approach provides objectivity to the estimation and reduces the bias.

We now apply the methodology proposed in Section 2 to three different data sets on (1) corporate obligors based in Eastern Europe, (2) corporate obligors based in Poland, and (3) retail obligors based in Germany. We use data on corporates from Poland and Eastern Europe because these are developing markets where the relations between the risk factors and the default event are not yet well investigated. In a developing market, the group of default drivers could be significantly different from what is observed in a developed market. By using data from developing markets, we try to find out whether the default drivers in these markets are significantly different from the default drivers in developed markets. Finally, we examine retail data from a developed market to check whether the default identification of our proposed algorithm is adequate on a data set that is not corporate.

3.1. *East-European data set*

The data set contains information for 7996 observations on 33 independent variables (covariates or features) and on 1 binary target variable which indicates whether a default occurred one year after the issue of the financial statement. The 33 covariates are constructed based on data from the entity's financial statements. These financial ratios are split into several groups and further analysed. For the feature names and construction refer to Appendix 1, table A1. The data are on an annual basis from the period 2007 to 2012. The data set is not publicly available, but the authors can share the data set if requested.

3.2. *Polish data set*

This data set is publicly available (Tomczak 2016) and was collected from Emerging Markets Information Service, which is a database containing information on emerging markets around the world. The bankrupt companies are analysed in the period 2000–2012, while the still operating companies are evaluated from 2007 to 2013. The data set has 5910 observations on 64 independent variables. The default indicator shows the bankruptcy status after 1 year. For the feature names and construction, refer to Appendix 1, table A2.

3.3. *German data set*

Also this data set is freely available (Hofmann 1994). It contains retail data for German credit borrowers with 1000 observations on 20 independent variables (covariates or features) and on 1 binary target variable which indicates the presence of default. The data set contains categorical and numerical variables. Following Agresti (2019), who explains that the choice of scores for the categorical variables has little impact on the final result, for clarity and simplicity we transform the categorical variables on a numerical scale by mapping them to integer numbers corresponding to the level

of each category. For example, a categorical variable with categories small, medium and large is mapped to an integer variable with values 1, 2 and 3, respectively. After that all the variables (continuous and categorical) are standardized. There are no missing values. For the feature names and construction refer to Appendix 1, table A3.

3.4. Feature selection

The literature offers a variety of algorithms for variable selection such as filter and wrapper methods. However, the main goal of our analysis is to examine the effect of Bayesian regularization on ANNs. Therefore, we apply a simple approach to variable selection based on the 80% percentile of the vector containing the absolute value of the correlation with the target variable. We select only variables whose correlation is equal or above the 80% percentile of the vector containing the absolute value of the correlation with the target variable. This leads to a balanced number of variables that are shown in table 1. Nonetheless, not to bias our results based on a single combination of variables, we report our results for different number of variables by changing the percentile value from 0% to 90%, see Appendix 2. This is consistent with the principle applied in Sariev and Germano (2019), where model performance is assessed comparing a model on a different set of variables.

3.5. Results

Table 2 presents eight different feed-forward neural network architectures. Prior to applying the networks on the data, we need to make a choice on the number of neurons and the number of layers for each network. Determining the number of neurons and layers is driven by many factors such as the number of variables in the model, the number of data points, etc. In order to avoid reporting biased results, we run each network on a range of different combinations of layers and neurons. This allows us to monitor the performance of the network over different network architectures and to summarize the network performance. Although there is no clear rule on selecting the number of neurons and layers, we follow Demuth *et al.* (2014) who argue that the number of neurons should be lower than the number of variables used in the network. Furthermore, the number of hidden layers should not be more than two to three because most problems are tackled even with one hidden layer. Adding many hidden layers on small data sets (less than one million observations) does not result in a better performance. Therefore, we decide to report the performance of the network on a combination of neurons that range from 1 to 25 and hidden layers that range from 1 to 3. We investigate combinations with more neurons than Demuth *et al.* (2014) suggested so that our results are more comprehensive. However, we show that increasing the number of layers does not lead to a higher performance; see Section 3.6.

We acknowledge that our decision on the number of neurons and layers is subjective, but we aim to cover a wide enough range of neurons and layers so that our results are less biased than using just a single combination of hidden layers and neurons. For classical regularization, the regularization parameter should be determined before applying the network to the data. Based on 10-fold cross-validation, we estimated the parameter γ for classical regularization, see equation (7),

Table 1. Selected variables by data set based on the 80% percentile of the correlation to the target variable.

Data set	Selected variables
East-European data	payables turnover, return on assets, cash ratio, income from sales/total assets, liquid assets/total assets, interest coverage
Polish data	total costs/total sales, (sales − cost of products sold)/sales, profit on sales/sales, working capital, logarithm of total assets, sales(n)/sales($n-1$), sales/inventory, working capital/total assets, sales/receivables, short-term liabilities/total assets, total liabilities/total assets, sales/total assets
German data	duration in months of the account, credit history, checking account status

Table 2. Performance of the ANN on the East-European; Polish and German test data when using factors based on the 80% percentile of the correlation to the target variable.

Architecture	Regularization	ES	Correct	Good	Bad	Gini	CPU time/s
			East-European data				
1	No	No	0.66	0.59	0.73	0.59	1.2
2	No	Yes	0.67	0.58	0.75	0.61	0.9
3	Classical	No	0.66	0.58	0.73	0.59	0.9
4	Classical	Yes	0.67	0.58	0.75	0.60	0.9
5	Bayesian	No	0.66	0.59	0.73	0.55	1.6
6	Bayesian	Yes	0.67	0.73	0.62	0.50	1.7
7	Bayesian MCMC	No	0.71	0.69	0.74	0.61	19.1
8	Bayesian MCMC	Yes	0.70	0.70	0.70	0.57	18.3
			Polish data				
1	No	No	0.67	0.75	0.57	0.52	0.8
2	No	Yes	0.65	0.75	0.56	0.52	0.8
3	Classical	No	0.67	0.79	0.54	0.53	0.9
4	Classical	Yes	0.65	0.76	0.55	0.52	0.8
5	Bayesian	No	0.63	0.68	0.52	0.55	1.5
6	Bayesian	Yes	0.64	0.87	0.39	0.53	1.6
7	Bayesian MCMC	No	0.68	0.75	0.64	0.52	17.1
8	Bayesian MCMC	Yes	0.68	0.66	0.69	0.56	16.4
			German data				
1	No	No	0.68	0.63	0.72	0.60	1.2
2	No	Yes	0.67	0.63	0.71	0.61	1.0
3	Classical	No	0.68	0.61	0.74	0.61	1.3
4	Classical	Yes	0.67	0.61	0.74	0.61	1.0
5	Bayesian	No	0.66	0.67	0.65	0.57	2.0
6	Bayesian	Yes	0.61	0.43	0.75	0.55	2.3
7	Bayesian MCMC	No	0.68	0.65	0.70	0.57	20.6
8	Bayesian MCMC	Yes	0.70	0.72	0.67	0.58	19.4

Notes: Correct: the percentage of overall correctly classified obligors; Good: the percentage of correctly classified good obligors; Bad: the percentage of correctly classified bad obligors; Gini: the Gini coefficient; CPU time/s: the CPU time in seconds needed for one run of the network.

to be 0.05. All networks are trained with the MSE loss function. One third of the data is left for testing the networks, two-thirds of the data are allocated for training and validating.

Following the above logic, we report our results in table 2. The first column shows the architecture type; the second column shows the regularization type: classical, Bayesian or Bayesian based on MCMC; the third column indicates whether ES was applied or not. The fourth to seventh columns give the mean percentage of correctly classified observations, the percentage of non-defaulted obligors, the percentage of defaulted obligors and the Gini coefficient on the grid of 1–25 neurons and 1–3 hidden layers. All results are reported on test data. Finally, the eighth column presents the average CPU time in seconds to compute a network; the CPU was an Intel Celeron N2840 with 2.16 GHz.

Based on table 2, we observe that:

(i) The percentage of overall correctly classified obligors is the highest for a network architecture where the regularization parameters are estimated by the Bayesian approach with the proposed MCMC estimation rather than the Gaussian approximation.
(ii) In some cases, the ES procedure can lead to a better performance but in other cases ES undermines the network performance.
(iii) The computational time needed for the MCMC estimation of the network is significantly higher than for the other networks but the Bayesian estimation automatically estimates the regularization thus reducing the bias.

The above observations are valid for all data sets. Below we examine the results for each data set separately.

(i) For the East-European data Bayesian regularization with MCMC leads to the highest overall performance. The improvement in performance is 4% which is high enough to make a difference from a practical point of view. In terms of identification of bad obligors and Gini coefficient, Bayesian regularization with MCMC performs similarly to classical regularization.
(ii) For the Polish data Bayesian regularization with MCMC leads to the highest overall performance. The improvement in performance is 1%, which can be ignored for practical purposes. However, in terms of identification of bad obligors and Gini coefficient, Bayesian regularization with MCMC performs significantly better than the other methods.
(iii) For the German data, Bayesian regularization with MCMC leads to the highest overall performance. The improvement in performance is 2%, which may make a difference in a situation where overall performance is of utmost importance. However, in terms of identification of bad obligors and Gini coefficient, Baeysian regularization with MCMC under-performs compared to the other methods.

The results in table 2 are based on the 80% percentile of the correlation to the target variable. In Appendix 2, one can see the results for the other combinations of variables but the conclusion stays the same. In all cases, Bayesian regularization with MCMC overall outperforms the other methods.

Finally, in table A4 in Appendix 2, we apply a two-sample *t*-test on the overall performance of our proposed method namely architectures 7 and 8. The two-sample *t*-test is a parametric test that compares the location parameter of two independent data samples. The statistics of the test follows a Student's *t* distribution. The null hypothesis states that the means of two populations are equal. In table A4, 1 indicates a rejection of the null hypothesis. Therefore, we can claim that our results are statistically different for each data set.

Figure 1 presents distributions of the overall correctly classified obligors per data set and per network architecture that are shown in table 2. We can see from figure 1 that the distribution of the overall correctly classified obligors for architectures 7 and 8 is right skewed for the East-European and Polish data. The results in figure 1 are based on the 80% percentile of the correlation to the target variable. In Appendix 2 one can see the results for the other combinations of variables but the conclusion stays the same. In all cases, Bayesian regularization with MCMC overall outperforms the other methods.

3.6. *Neural network performance on increasing the number of layers*

Inspired by the flexibility of the deep neural network paradigm, we tried to increase the number of layers with the goal of increasing the performance on the test data. However, contrary to our expectations, the generalization power of the network decreased for each data set, as can be seen from figure 2. The decrease in performance is different for each data set. For the Polish and German, the decrease of performance is not significant but for the East-European data the decrease is significant. The reason is that our data sets are not big enough to allow the application of many layers. The networks with more than 4 layers significantly overfit the data.

3.7. *Comparison to other classification algorithms*

The main objective of our research is to analyse the effect of Bayesian regularization and compare it to classical regularization for ANNs. We report our results as an average over different combinations of layers and neurons. Therefore, we do not report the maximum classification accuracy that can be achieved rather we aim to present the effect of Bayesian regularization with MCMC over different network architectures and advocate that on average our proposed approach leads to higher performance when compared to other regularization approaches for ANNs. However, for the purpose of completeness, we apply two other non-linear classification methods to the three data sets. The first is a support vector machine (SVM) and the second is *k*-nearest neighbours (KNN). The results in terms of classification accuracy are shown in table 3. Overall the performance is similar to our proposed method. On the Polish corporate data SVM and KNN outperform but as we emphasized before the results we report in table 2 are averaged over a grid of different neurons and layers and therefore are not directly comparable to the results in table 3. Therefore, the performance reported in table 2 is not the

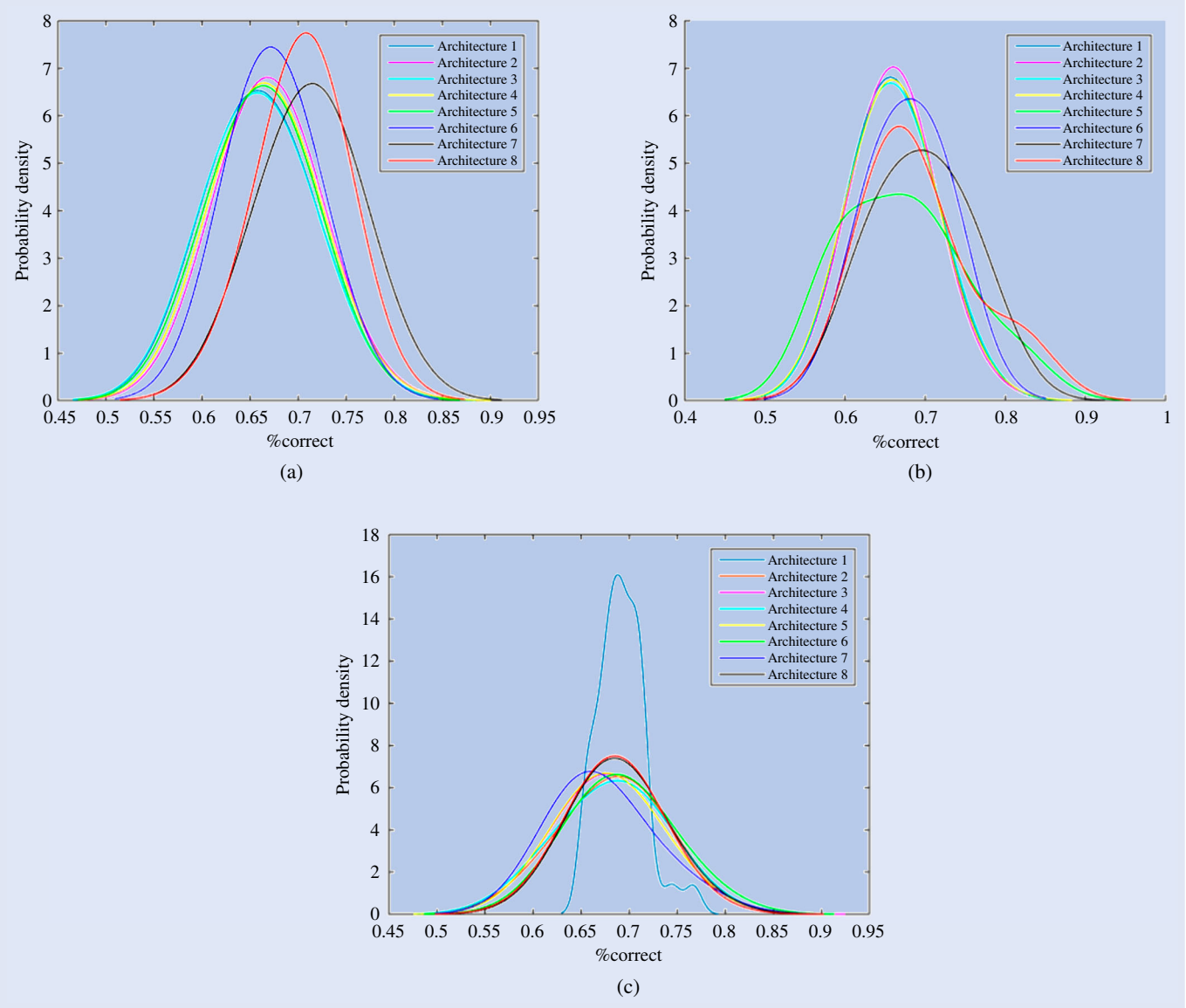

Figure 1. Distribution of the overall correctly classified obligors for the East-European (a), Polish (b) and German (c) data. Results are based on the 80% percentile of the correlation to the target variable.

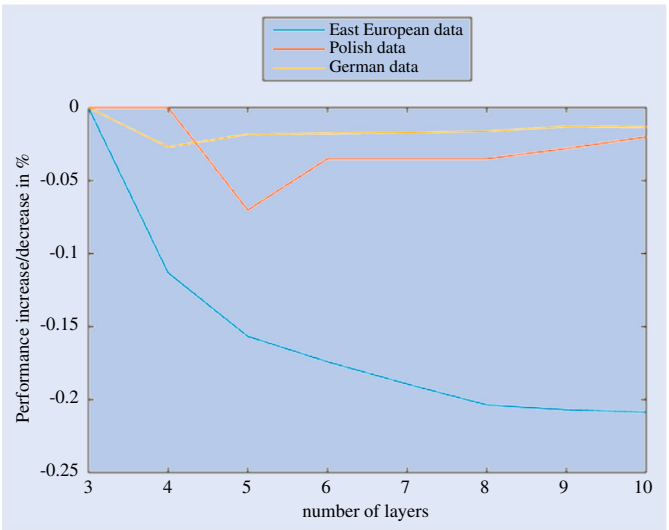

Figure 2. Percentage increase/decrease per number of layers for each data set.

Table 3. Overall accuracy (per cent of classified obligors) by SVM and KNN.

	East-European data		Polish data		German data	
Percentile	SVM	KNN	SVM	KNN	SVM	KNN
0%	0.62	0.63	0.70	0.71	0.49	0.70
50%	0.65	0.63	0.73	0.74	0.63	0.62
60%	0.69	0.63	0.69	0.73	0.69	0.67
70%	0.67	0.62	0.69	0.73	0.71	0.63
80%	0.66	0.62	0.72	0.74	0.65	0.57
90%	0.66	0.61	0.72	0.69	0.62	0.65

Notes: The results are shown per variable selection combination based on the 0%, 50%, 60%, 70%, 80% and 90% percentile of the correlation to the target variable.

highest that could be achieved using Bayesian regularization but this average performance is close to the maximum performance we achieve when we apply SVM and KNN to the data.

4. Policy implications

Identifying a classification method to estimate the PD is an important factor but equally important is deriving business intuition from the selected default factors. Typically, PD models are used by a non-technical audience and the interpretation of the default factors from an industry prospective is of utmost importance. For that reason, we split the selected ratios into three categories†

(i) Leverage category: ratios that signal how much debt and debt-related costs a company utilizes against its equity or assets. Effectively this category indicates the level of indebtedness of a company.
(ii) Profitability category: ratios that signal the ability of a company to generate income relative to its equity or assets. Effectively this category indicates how efficiently a company utilizes it assets.
(iii) Liquidity category: ratios that signal a company's ability to meet the current liabilities when they become due with its current assets. Effectively this category indicates the ability of a company to pay off its short-term obligations.

Table 4 presents the allocation of the selected ratios from the variable selection method on the two corporate data sets (East-European and Polish) to each of the above three categories.

As can be seen from table 4, the default risk in the Polish data set is driven mainly by the profitability ratios, followed by the leverage ratios. Liquidity ratios do not play an important role in determining the default risk of the Polish obligors. We compare our approach with that of Liang *et al.* (2016) where they split the financial ratios into several groups namely: solvency (leverage) ratios, profitability ratios, capital structure ratios, cash flow ratios, ownership ratios, turnover ratios, and growth ratios. They found that leverage and profitability ratios are the most important categories in identifying defaults. Interestingly, they have used data from the Taiwan Stock Exchange. The fact that their findings align with ours proofs the significance and the universality of the leverage and profitability ratios. Another study by Al-Kassar and Soileau (2014) also indicates the importance of profitability and leverage ratios through the use of factor analysis. However, they advocate that non-financial data are also important in identifying and measuring default risk. Furthermore, a study by Chen *et al.* (2011) emphasizes the role of the profitability and leverage ratios. The analysis is done on 20 000 solvent and 1000 insolvent companies. Their study applies SVM on German companies and shows the importance of profitability and leverage in identifying defaults.

Similarly the default risk in the East-European data set is driven by profitability and leverage ratios but it is also driven by liquidity ratios. We compare our approach with that of Marilena and Alina (2015) where liquidity and leverage ratios are identified as a major default driver. Their work applies multiple discriminant analysis, logistic regression and ANNs. The data used are from the Bucharest Stock Exchange principal market. Moreover, a study by Tian *et al.* (2015) also indicates the importance of liquidity and leverage ratios. They use North American financial data on corporate obligors and apply the LASSO method for variable selection. Finally, Tian *et al.* (2015) claim that their approach is superior to the one given by the popular distance to default model proposed by Merton (1974).

Table 4. Selected ratios based on the 80% percentile of the correlation to the target, allocated into three main financial categories: leverage, profitability and liquidity on the East-European (E) and on the Polish (P) data.

Category	Ratio
Leverage ratios	interest coverage (E), short-term liabilities/total assets (P), total liabilities/total assets (P), total costs/total sales (P)
Profitability ratios	return on assets (E), income from sales/total assets (E), sales/total assets (P), sales/inventory (P), sales/receivables (P), sales(n)/sales($n-1$) (P), profit on sales/sales (P), (sales − cost of products sold)/sales (P)
Liquidity ratios	cash ratio (E), liquid assets/total assets (E), working capital (P)

We note that the major difference in default drivers between the East-European data and the Polish data is the higher importance of liquidity ratios for the former, the reason being that Polish companies are in general bigger and liquidity is not a major indication of default risk. Practically, larger companies have access to cheaper funding, whereas smaller companies incur higher funding costs.

Due to the low number of selected features in the German retail data set, we are not able to allocate them into different groups. However, most of the variables in the German retail data are based on the status and duration of the current account. This is aligned with the study of Barrell *et al.* (2010), which shows evidence that the status of the current account is a major predictor of mortgage defaults.

5. Discussion and conclusions

In this paper, we propose an improvement of a Bayesian approach to regularize feed-forward neural networks. The Bayesian approach is attractive because it provides automatic determination of the regularization parameters. Moreover, we demonstrate that the improved Bayesian approach performs well when compared to the classical regularization approach for neural networks. We find that using a MCMC scheme to estimate the Bayesian regularization parameters leads to higher performance than using a Gauss–Newton approximation. Furthermore, the application of early stopping on the network does not guarantee higher performance.

We analysed three data sets; two are corporate and one retail. From a policy prospective, three groups of financial ratios are identified as major drivers of default risk: profitability ratios, leverage ratios and liquidity ratios. The effect of

† Payables turnover = supply payables × 360/cost of goods sold (East-European data) and working capital/total assets as well as logarithm of total assets (Polish data) cannot be allocated to these three groups.

liquidity ratios is higher on the East-European data and the effect of profitability ratios is higher on the Polish data.

The findings of this paper yield promising insights into the potential of Bayesian regularization to efficiently estimate the network weights. Practically, this leads to making better informed and less biased credit risk decisions.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

The funding of the Systemic Risk Centre by the Economic and Social Research Council (ESRC) is gratefully acknowledged [grant number ES/K002309/1].

References

Addo, P.M., Guégan, D. and Hassani, B., Credit risk analysis using machine and deep learning models. Technical report 18003, Université Panthéon-Sorbonne (Paris 1), Centre d'Economie de la Sorbonne, 2018.

Agresti, A., *An Introduction to Categorical Data Analysis*, 3rd edn, chap. 2, pp. 25–56, 2019 (Wiley: Hoboken, NJ).

Al-Kassar, T.A. and Soileau, J.S., Financial performance evaluation and bankruptcy prediction (failure). *Arab Econ. Bus. J.*, 2014, **9**, 147–155.

Ashiquzzaman, A., Tushar, A.K., Islam, M.R. and Kim, J., Reduction of overfitting in diabetes prediction using deep learning neural network, 2017. arXiv/1707.08386.

Barrell, R., Davis, E., Karim, D. and Liadze, I., The impact of global imbalances: Does the current account balance help to predict banking crises in OECD countries? NIESR Discussion Paper 351, 2010.

Bell, J., *Machine Learning*, chap. 5, pp. 91–117, 2015 (Wiley: Indianapolis, IN).

Bonini, S. and Caivano, G., Probability of default modeling: A machine learning approach. In *Proceedings of the Mathematical and Statistical Methods for Actuarial Sciences and Finance*, edited by M. Corazza, M. Durbàn, A. Grané, C. Perna and M. Sibillo, pp. 173–177, 2018 (Springer: Cham).

Chen, S., Härdle, W.K. and Moro, R.A., Modeling default risk with support vector machines. *Quant. Finance*, 2011, **11**, 135–154.

Demuth, H.B., Beale, M.H., De Jess, O. and Hagan, M.T., *Neural Network Design*, 2nd ed., 2014 (Martin Hagan).

Deng, W., Smirnov, E., Timoshenko, D. and Andrianov, S., Comparison of regularization methods for imagenet classification with deep convolutional neural networks. *AASRI Procedia*, 2014, **6**, 89–94.

Dreyfus, S., Artificial neural networks, back propagation, and the Kelley-Bryson gradient procedure. *J. Guidance Control Dyn.*, 1990, **13**, 926–928.

Durand, D., *Risk Elements in Consumer Installment Financing*, pp. 189–201, 1941 (Cambridge, MA: National Bureau of Economy Research).

Farhadi, F., Learning activation functions in deep neural networks. Diploma Thesis, Department of Mathematics, Polytechnique Montréal, Montreal, Quebec, 2017.

Foresee, F. and Hagan, M., Gauss-Newton approximation to Bayesian learning. In *Proceedings of the 1997 International Joint Conference on Neural Networks*, Vol. 3, pp. 1930–1935, 1997 (IEEE: New York, NY).

Gelfand, A. and Smith, A., Sampling-based approaches to calculating marginal densities. *J. Am. Stat. Assoc.*, 1990, **85**, 398–409.

Gill, P. and Murray, W., Algorithms for the solution of the nonlinear least-squares problem. *SIAM J. Numer. Anal.*, 1978, **15**, 977–992.

Heaton, J.B., Polson, N.G. and Witte, J.H., Deep learning in finance, 2016. arXiv/1602.06561.

Hindi, K.E. and Al-Akhras, M., Neural network world. *Int. Inform. Inst.*, 2011, **21**, 311–325.

Hofmann, H., Statlog German credit data set, 1994. Available online at: https://archive.ics.uci.edu/ml/datasets/Statlog + (German + credit + data).

Kalayci, S., Kamasak, M. and Arslan S., Credit risk analysis using machine learning algorithms. *Proceedings of the 26th Signal Processing and Communications Applications Conference (SIU 2018)*, pp. 821–824, 2018 (IEEE/Curran Associates: Red Hook, NY).

Kim, H., Jung, S., Kim, T. and Park, K., Fast learning method for back-propagation neural network by evolutionary adaptation of learning rates. *Neurocomputing*, 1996, **11**, 101–106.

Lampinen, J. and Vehtari, A., Bayesian approach for neural networks—Reviews and case studies. *Neural Netw.*, 2001, **14**, 257–274.

Liang, D., Lu, C.C., Tsai, C.F. and Shih, G.A., Financial ratios and corporate governance indicators in bankruptcy prediction: A comprehensive study. *Eur. J. Oper. Res.*, 2016, **252**, 561–572.

MacKay, D., Bayesian interpolation. *Neural Comput.*, 1992, **4**, 415–447.

Marilena, M. and Alina, T., The significance of financial and non-financial information in insolvency risk detection. *Procedia Econ. Finance*, 2015, **26**, 750–756.

Merton, R.C., On the pricing of corporate debt: The risk structure of interest rates. *J. Finance*, 1974, **29**, 449–470.

Neal, R., Bayesian training of backpropagation networks by the hybrid Monte Carlo method. Technical report CRG-TR-92-1, Department of Computer Science, University of Toronto, 1992.

Neal, R., Bayesian learning for neural networks. PhD Thesis, University of Toronto, Toronto, Canada, 1996.

Nicolae-Eugen, C., Lowering evolved artificial neural network overfitting through high-probability mutation. In *Proceedings of the 2016 18th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, September, pp. 325–329, 2016.

Pérez-Martín, A., Pérez-Torregrosa, A. and Vaca, M., Big data techniques to measure credit banking risk in home equity loans. *J. Bus. Res.*, 2018, **89**, 448–454.

Rasmussen, C., A practical Monte Carlo implementation of Bayesian learning. In *Proceedings of the Advances in Neural Information Processing Systems 8*, edited by D. Touretzky, M. Mozer and M. Hasselmo, pp. 598–604, 1996 (MIT Press: Cambridge, MA).

Sariev, E. and Germano, G., An innovative feature selection method for support vector machines and its test on the estimation of the credit risk of default. *Rev. Financ. Econ.*, 2019, **37**, 404–427.

Specht, D., Probabilistic neural networks. *Neural Netw.*, 1990, **3**, 109–118.

Stallkamp, J., Schlipsing, M., Salmen, J. and Igel, C., Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Netw.*, 2012, **32**, 323–332.

Tian, S., Yu, Y. and Guo, H., Variable selection and corporate bankruptcy forecasts. *J. Bank. Finance*, 2015, **52**, 89–100.

Titterington, D., Bayesian methods for neural networks and related models. *Stat. Sci.*, 2004, **19**, 128–139.

Tomczak, S., Polish companies bankruptcy data set, 2016. Available online at: https://archive.ics.uci.edu/ml/datasets/Polish + company + bankruptcy + data.

Tufféry, S., *Data Mining and Statistics for Decision Making*, chap. 1, pp. 1–23, 2011 (Wiley: Chichester).

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y. and Manzagol, P., Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 2010, **10**, 3371–3408.

Wang, J. and Peng, C., A novel self-creating neural network for learning vector quantization. *Neural Process. Lett.*, 2000, **11**, 139–151.

Zhang, C., Vinyals, O., Munos, R. and Bengio, S., A study on overfitting in deep reinforcement learning, 2018. arXiv/1804.06893.

Appendices

Appendix 1

Table A1. Summary statistics on all ratios, East-European data.

Ratio name	Mean	Median
return on assets (ROA)	0.13	0.08
ROA before financial expenses	0.18	0.12
return on operating income	− 0.07	0.08
return on sales income	− 0.01	0.11
return on investment	0.06	0.03
cash ratio	0.45	0.01
quick ratio	2.06	0.50
operating cash flow ratio	4.21	1.16
liquid assets over total assets	0.03	0.00
working capital over total assets	0.49	0.48
financial autonomy	14.25	20.34
total funding ratio	0.76	0.83
long term funding ratio	0.20	0.39
total financial liabilities over total assets	0.39	0.22
supply payables over total assets	0.16	0.09
financial liabilities over total liabilities	0.39	0.35
equity over total liabilities	2.04	0.29
short term funding ratio	0.62	0.68
total liabilities coverage	1.37	0.17
financial liabilities coverage	11.45	0.41
current financial liabilities coverage	7.30	0.87
interest coverage	99.86	4.43
income from sales/total assets	1.73	1.00
employees' expense / sales income	0.13	0.06
earnings on operating income	1.10	0.94
payables turnover	263.44	39.30
inventory turnover	251.29	66.41
receivables turnover	97.17	20.27
total sales income	5348.68	524.00
total assets	3365.22	531.00
relative annual change in total sales	4.62	0.14
relative annual change in total assets	1.34	0.15
relative annual change in profit from main activities	11.18	− 0.08
absolute annual change in total liabilities	0.03	0.00

Table A2. Summary statistics on all ratios, Polish corporate data.

Ratio name	Mean	Median
net profit/total assets	− 0.02	0.05
total liabilities/total assets	0.47	0.45
working capital/total assets	0.19	0.22
current assets/short-term liabilities	5.00	4.89
(cash+short-term securities+receivables-short-term liabilities)/(operating expenses- depreciation) × 365	19.41	0.57
retained earnings/total assets	0.02	0.00
EBIT/total assets	− 0.11	0.06
book value of equity/total liabilities	5.74	1.16
sales/total assets	1.59	1.14
equity/total assets	0.55	0.52
(gross profit+extraordinary items+financial expenses)/total assets	0.07	0.00
gross profit/short-term liabilities	1.07	0.17
(gross profit + depreciation)/sales	0.35	0.07
gross profit + interest)/total assets	− 0.11	0.06
(total liabilities × 365)/(gross profit + depreciation)	1033.62	875.25
(gross profit + depreciation)/total liabilities	1.19	0.24
total assets/total liabilities	6.83	2.21
gross profit/total assets	− 0.10	0.06
gross profit/sales	− 0.09	0.04
(inventory × 365)/sales	56.67	38.62
sales(n)/sales($n-1$)	2.46	1.12
profit on operating activities/total assets	− 0.00	0.06
net profit/sales	− 0.10	0.03
gross profit(in 3 years)/total assets	0.14	0.16
(equity – share capital)/total assets	0.38	0.42
(net profit + depreciation)/total liabilities	1.09	0.21
profit on operating activities/financial expenses	463.64	1.15
working capital/fixed assets	10.23	0.55
logarithm of total assets	4.15	4.17
(total liabilities,cash)/sales	0.85	0.22
(gross profit + interest)/sales	− 0.07	0.04
(current liabilities × 365)/cost of products sold	2111.59	81.91
operating expenses/short-term liabilities	8.34	4.50
operating expenses/total liabilities	5.01	1.72
profit on sales/total assets	− 0.01	0.06
total sales/total assets	2.05	1.56
(current assets,inventories)/long-term liabilities	67.02	5.00
constant capital/total assets	0.65	0.62
profit on sales/sales	0.02	0.04
(current assets,inventory/receivables)/short-term liabilities	2.21	0.18
total liabilities/((profit on operating activities + depreciation) × 12/365)	2.19	0.09
profit on operating activities/sales	− 0.02	0.04
rotation receivables+inventory turnover in days	155.56	106.41
(receivables × 365)/sales	98.88	58.79
net profit/inventory	66.63	0.29
(current assets – inventory)/short-term liabilities	4.01	1.07
(inventory × 365)/cost of products sold	137.42	42.351
EBITDA/total assets	− 0.09	0.02
EBITDA/sales	− 0.07	0.01
current assets/total liabilities	4.17	1.29
short-term liabilities/total assets	0.43	0.33
(short-term liabilities × 365)/cost of products sold)	0.73	0.22
equity/fixed assets	11.20	1.30
constant capital/fixed assets	12.11	1.45
working capital	10817.31	1802.80
(sales, cost of products sold)/sales	0.06	0.05
(current assets – inventory – short-term liabilities)/(sales – gross profit – depreciation)	− 0.26	0.11
total costs/total sales	0.96	0.950
long-term liabilities/equity	0.28	0.01
sales/inventory	911.03	9.45
sales/receivables	10.94	6.21
(short-term liabilities × 365)/sales	241.98	73.78
sales/short-term liabilities	9.13	4.94
sales/fixed assets	65.28	4.22

Table A3. Summary statistics for all ratios, German retail data.

Ratio number and name	Median	Mean
1 status of existing checking account	2	2.58
2 duration in months of the account	18	20.9
3 credit history	3	3.6
4 credit purpose	2	2.9
5 credit amount	2320	3271
6 savings account/bonds	1	2.1
7 present employment since	3	3.9
8 installment rate in percentage of disposable income	3	2.973
9 personal status and sex	3	2.7
10 other debtors/guarantors	1	1.2
11 present residence since	3	2.845
12 property indicator	2	2.4
13 age in years	33	35.55
14 other installment plans	3	2.7
15 housing indicator	2	1.9
16 number of existing credits at this bank	1	1.41
17 job status	3	2.9
18 number of people being liable to provide maintenance for	1	1.2
19 telephone availability	1	1.4
20 foreign worker indicator	1	1

Notes: The median and the mean are shown before standardization of the variable.

Appendix 2

Table A5. Performance of the ANN on the East-European, Polish and German test data when using factors based on the 90% percentile of the correlation to the target variable.

Architecture	Regularization	ES	Correct	Good	Bad	Gini
		East-European data				
1	No	No	0.67	0.56	0.77	0.58
2	No	Yes	0.67	0.57	0.77	0.60
3	Classical	No	0.67	0.56	0.77	0.58
4	Classical	Yes	0.67	0.57	0.77	0.60
5	Bayesian	No	0.66	0.54	0.77	0.59
6	Bayesian	Yes	0.66	0.51	0.80	0.61
7	Bayesian MCMC	No	0.67	0.58	0.77	0.62
8	Bayesian MCMC	Yes	0.68	0.66	0.70	0.58
		Polish data				
1	No	No	0.67	0.73	0.60	0.52
2	No	Yes	0.67	0.66	0.67	0.59
3	Classical	No	0.67	0.70	0.64	0.52
4	Classical	Yes	0.67	0.68	0.66	0.59
5	Bayesian	No	0.65	0.82	0.46	0.54
6	Bayesian	Yes	0.68	0.66	0.70	0.56
7	Bayesian MCMC	No	0.74	0.74	0.71	0.61
8	Bayesian MCMC	Yes	0.74	0.73	0.75	0.66
		German data				
1	No	No	0.68	0.63	0.72	0.60
2	No	Yes	0.67	0.63	0.71	0.61
3	Classical	No	0.68	0.61	0.74	0.61
4	Classical	Yes	0.67	0.61	0.74	0.61
5	Bayesian	No	0.66	0.67	0.65	0.57
6	Bayesian	Yes	0.61	0.43	0.75	0.55
7	Bayesian MCMC	No	0.68	0.65	0.70	0.57
8	Bayesian MCMC	Yes	0.68	0.64	0.72	0.58

Notes: Correct: the percentage of overall correctly classified obligors; Good: the percentage of correctly classified good obligors; Bad: the percentage of correctly classified bad obligors; Gini: the Gini coefficient.

Table A4. Statistical significance of Bayesian MCMC (architectures 7 and 8) on the overall accuracy per data set: East-European data (E), Polish data (P), German data (G).

	E MCMC1	E MCMC2	P MCMC1	P MCMC2	G MCMC1	G MCMC2
Architecture	7	8	7	8	7	8
1	1	1	1	1	0	1
2	1	1	1	1	1	1
3	1	1	1	1	0	1
4	1	1	1	1	1	1
5	1	1	1	1	1	1
6	1	1	1	1	1	1
7	0	1	0	0	0	1
8	1	0	0	0	1	0

Notes: 1 indicates statistical difference, 0 indicates no statistical difference. MCMC1 and MCMC2 stand for Architectures 7 and 8 in table 2.

Table A6. Performance of the ANN on the East-European, Polish and German test data when using factors based on the 70% percentile of the correlation to the target variable.

Architecture	Regularization	ES	Correct	Good	Bad	Gini
		East-European data				
1	No	No	0.66	0.63	0.69	0.58
2	No	Yes	0.66	0.59	0.73	0.60
3	Classical	No	0.66	0.62	0.70	0.58
4	Classical	Yes	0.66	0.60	0.73	0.60
5	Bayesian	No	0.67	0.55	0.80	0.55
6	Bayesian	Yes	0.67	0.63	0.71	0.54
7	Bayesian MCMC	No	0.70	0.71	0.68	0.58
8	Bayesian MCMC	Yes	0.69	0.72	0.66	0.56
		Polish data				
1	No	No	0.66	0.78	0.55	0.51
2	No	Yes	0.64	0.76	0.53	0.54
3	Classical	No	0.66	0.76	0.57	0.50
4	Classical	Yes	0.65	0.77	0.52	0.54
5	Bayesian	No	0.66	0.71	0.59	0.53
6	Bayesian	Yes	0.65	0.56	0.66	0.49
7	Bayesian MCMC	No	0.69	0.73	0.62	0.51
8	Bayesian MCMC	Yes	0.67	0.67	0.66	0.54
		German data				
1	No	No	0.68	0.68	0.66	0.59
2	No	Yes	0.67	0.65	0.68	0.59
3	Classical	No	0.68	0.69	0.66	0.59
4	Classical	Yes	0.67	0.65	0.69	0.59
5	Bayesian	No	0.68	0.67	0.69	0.52
6	Bayesian	Yes	0.60	0.65	0.53	0.54
7	Bayesian MCMC	No	0.70	0.68	0.71	0.59
8	Bayesian MCMC	Yes	0.69	0.70	0.69	0.60

Notes: Correct: the percentage of overall correctly classified obligors; Good: the percentage of correctly classified good obligors; Bad: the percentage of correctly classified bad obligors; Gini: the Gini coefficient.

Table A7. Performance of the ANN on the East-European, Polish and German test data when using factors based on the 60% percentile of the correlation to the target variable.

Architecture	Regularization	ES	Correct	Good	Bad	Gini
		East-European data				
1	No	No	0.67	0.63	0.70	0.59
2	No	Yes	0.65	0.57	0.73	0.60
3	Classical	No	0.66	0.61	0.71	0.58
4	Classical	Yes	0.65	0.57	0.73	0.60
5	Bayesian	No	0.68	0.61	0.74	0.53
6	Bayesian	Yes	0.67	0.62	0.70	0.55
7	Bayesian MCMC	No	0.70	0.69	0.70	0.59
8	Bayesian MCMC	Yes	0.71	0.67	0.75	0.63
		Polish data				
1	No	No	0.66	0.74	0.57	0.51
2	No	Yes	0.62	0.79	0.45	0.53
3	Classical	No	0.66	0.76	0.56	0.51
4	Classical	Yes	0.63	0.82	0.42	0.54
5	Bayesian	No	0.63	0.64	0.59	0.53
6	Bayesian	Yes	0.63	0.59	0.64	0.48
7	Bayesian MCMC	No	0.67	0.72	0.64	0.53
8	Bayesian MCMC	Yes	0.66	0.68	0.64	0.52
		German data				
1	No	No	0.70	0.66	0.73	0.59
2	No	Yes	0.68	0.67	0.68	0.58
3	Classical	No	0.70	0.66	0.72	0.59
4	Classical	Yes	0.68	0.67	0.68	0.58
5	Bayesian	No	0.69	0.73	0.65	0.52
6	Bayesian	Yes	0.62	0.52	0.68	0.50
7	Bayesian MCMC	No	0.69	0.69	0.69	0.58
8	Bayesian MCMC	Yes	0.70	0.71	0.69	0.59

Notes: Correct: the percentage of overall correctly classified obligors; Good: the percentage of correctly classified good obligors; Bad: the percentage of correctly classified bad obligors; Gini: the Gini coefficient.

Table A8. Performance of the ANN on the East-European, Polish and German test data when using factors based on the 50% percentile of the correlation to the target variable.

Architecture	Regularization	ES	Correct	Good	Bad	Gini
		East-European data				
1	No	No	0.66	0.62	0.69	0.59
2	No	Yes	0.66	0.61	0.71	0.59
3	Classical	No	0.66	0.62	0.69	0.59
4	Classical	Yes	0.66	0.61	0.71	0.59
5	Bayesian	No	0.66	0.58	0.74	0.54
6	Bayesian	Yes	0.67	0.74	0.60	0.56
7	Bayesian MCMC	No	0.69	0.72	0.66	0.58
8	Bayesian MCMC	Yes	0.70	0.73	0.67	0.58
		Polish data				
1	No	No	0.68	0.78	0.57	0.51
2	No	Yes	0.64	0.73	0.54	0.54
3	Classical	No	0.67	0.77	0.57	0.51
4	Classical	Yes	0.64	0.75	0.53	0.54
5	Bayesian	No	0.64	0.45	0.81	0.53
6	Bayesian	Yes	0.65	0.57	0.73	0.49
7	Bayesian MCMC	No	0.69	0.74	0.56	0.52
8	Bayesian MCMC	Yes	0.67	0.67	0.68	0.53
		German data				
1	No	No	0.66	0.65	0.67	0.56
2	No	Yes	0.66	0.67	0.65	0.59
3	Classical	No	0.66	0.65	0.67	0.56
4	Classical	Yes	0.66	0.68	0.64	0.58
5	Bayesian	No	0.68	0.65	0.70	0.55
6	Bayesian	Yes	0.61	0.44	0.72	0.50
7	Bayesian MCMC	No	0.69	0.70	0.68	0.59
8	Bayesian MCMC	Yes	0.70	0.69	0.70	0.61

Notes: Correct: the percentage of overall correctly classified obligors; Good: the percentage of correctly classified good obligors; Bad: the percentage of correctly classified bad obligors; Gini: the Gini coefficient.

Table A9. Performance of the ANN on the East-European, Polish and German test data when using factors based on the 0% percentile of the correlation to the target variable.

Architecture	Regularization	ES	Correct	Good	Bad	Gini
		East-European data				
1	No	No	0.65	0.60	0.70	0.60
2	No	Yes	0.65	0.60	0.71	0.60
3	Classical	No	0.65	0.60	0.70	0.60
4	Classical	Yes	0.65	0.60	0.71	0.60
5	Bayesian	No	0.65	0.65	0.66	0.55
6	Bayesian	Yes	0.67	0.66	0.67	0.56
7	Bayesian MCMC	No	0.71	0.67	0.75	0.61
8	Bayesian MCMC	Yes	0.69	0.68	0.70	0.60
		Polish data				
1	No	No	0.65	0.76	0.52	0.51
2	No	Yes	0.62	0.74	0.49	0.54
3	Classical	No	0.66	0.80	0.50	0.51
4	Classical	Yes	0.62	0.78	0.44	0.54
5	Bayesian	No	0.63	0.67	0.55	0.51
6	Bayesian	Yes	0.64	0.87	0.39	0.53
7	Bayesian MCMC	No	0.67	0.79	0.56	0.52
8	Bayesian MCMC	Yes	0.64	0.65	0.60	0.51
		German data				
1	No	No	0.65	0.68	0.62	0.56
2	No	Yes	0.66	0.68	0.64	0.57
3	Classical	No	0.65	0.67	0.62	0.56
4	Classical	Yes	0.66	0.68	0.64	0.57
5	Bayesian	No	0.68	0.76	0.59	0.53
6	Bayesian	Yes	0.67	0.67	0.62	0.54
7	Bayesian MCMC	No	0.68	0.66	0.69	0.61
8	Bayesian MCMC	Yes	0.67	0.62	0.72	0.58

Notes: Correct: the percentage of overall correctly classified obligors; Good: the percentage of correctly classified good obligors; Bad: the percentage of correctly classified bad obligors; Gini: the Gini coefficient.

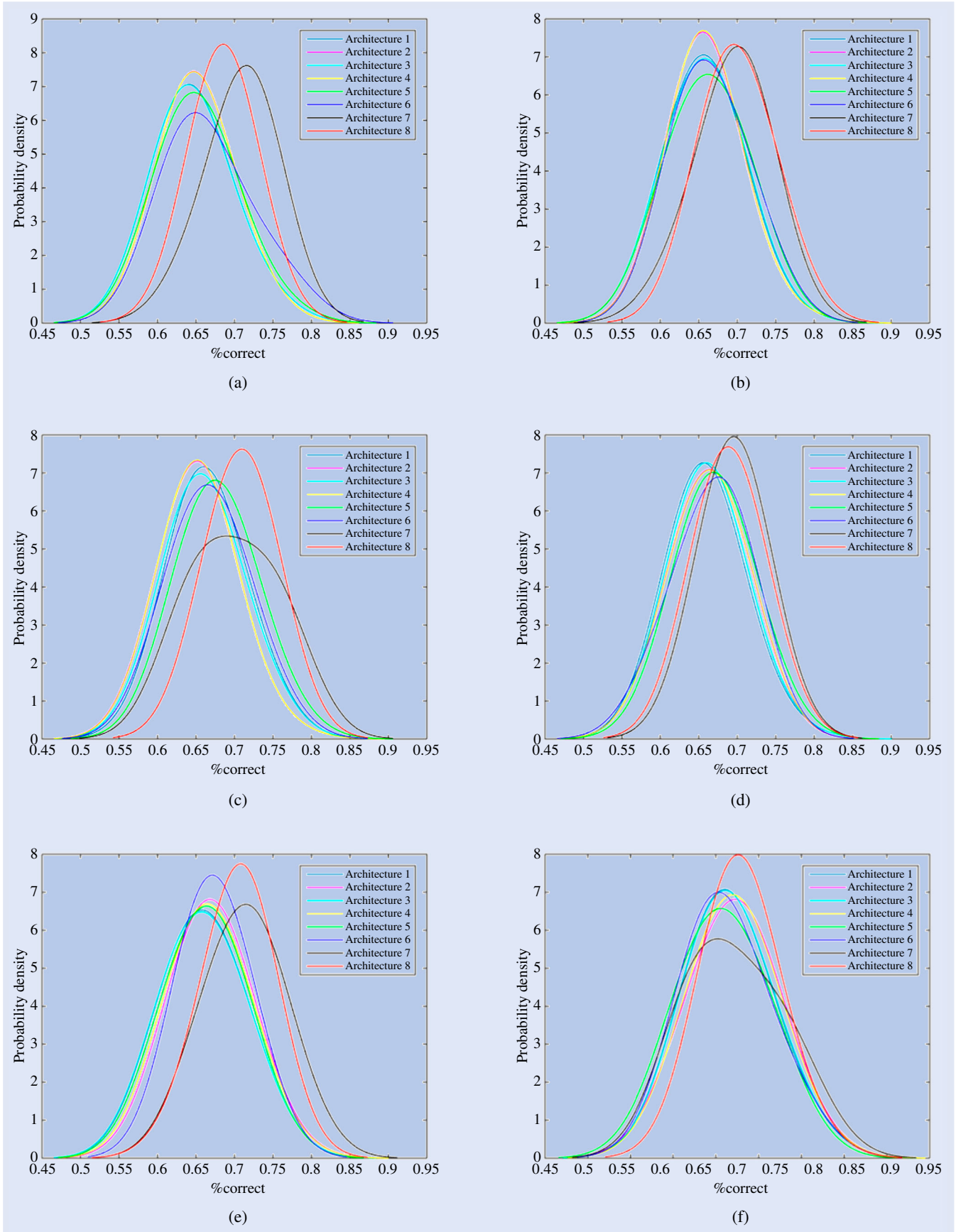

Figure A1. Clockwise, distribution of the overall correctly classified obligors for the East-European data. Results are based on the 0%, 50%, 60%, 70%, 80% and 90% percentile of the correlation to the target variable. (a) 0% percentile. (b) 50% percentile. (c) 60% percentile. (d) 70% percentile. (e) 80% percentile and (f) 90% percentile.

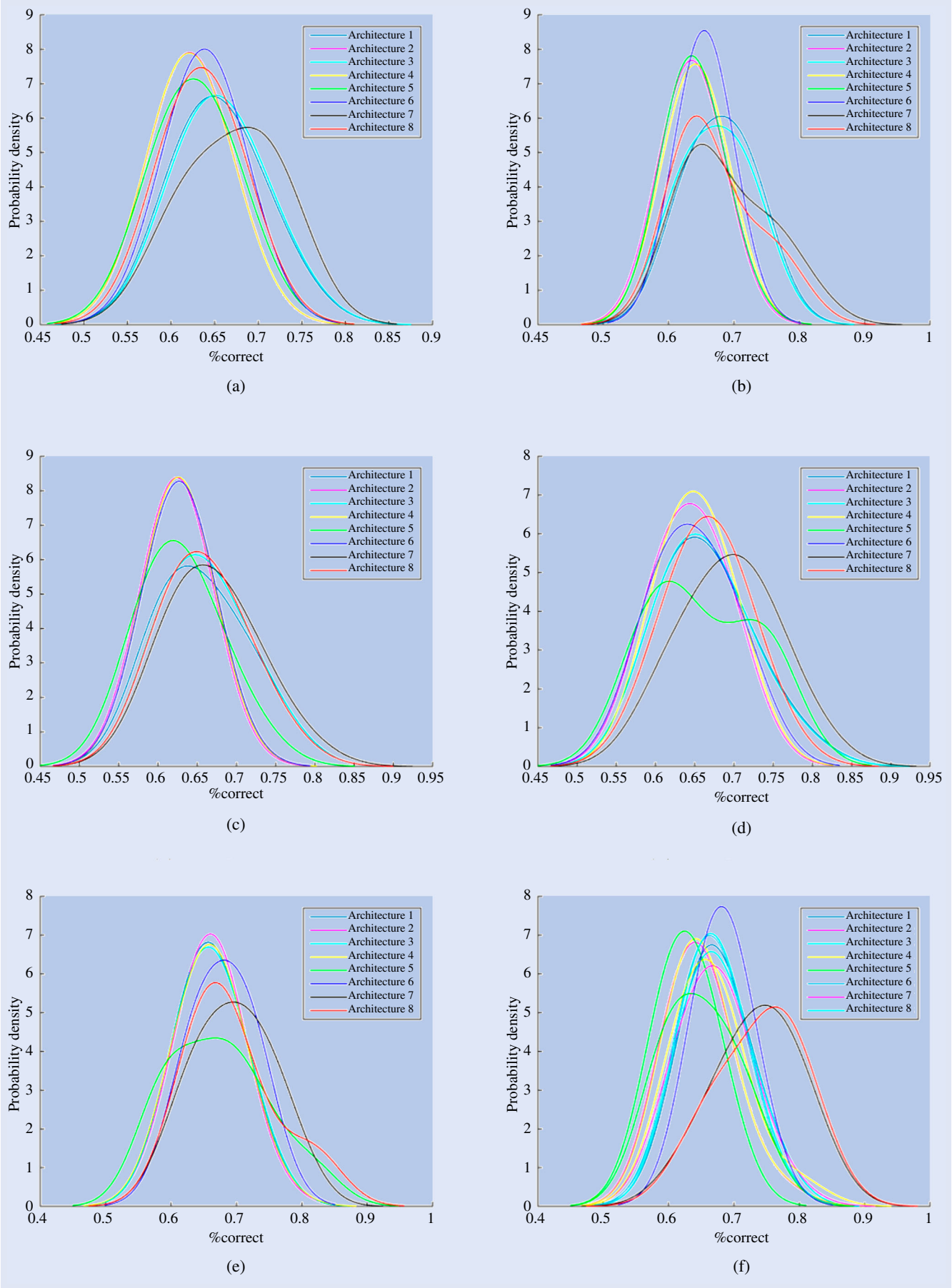

Figure A2. Clockwise, distribution of the overall correctly classified obligors for the Polish data. Results are based on the 0%, 50%, 60%, 70%, 80% and 90% percentile of the correlation to the target variable. (a) 0% percentile. (b) 50% percentile. (c) 60% percentile. (d) 70% percentile. (e) 80% percentile and (f) 90% percentile.

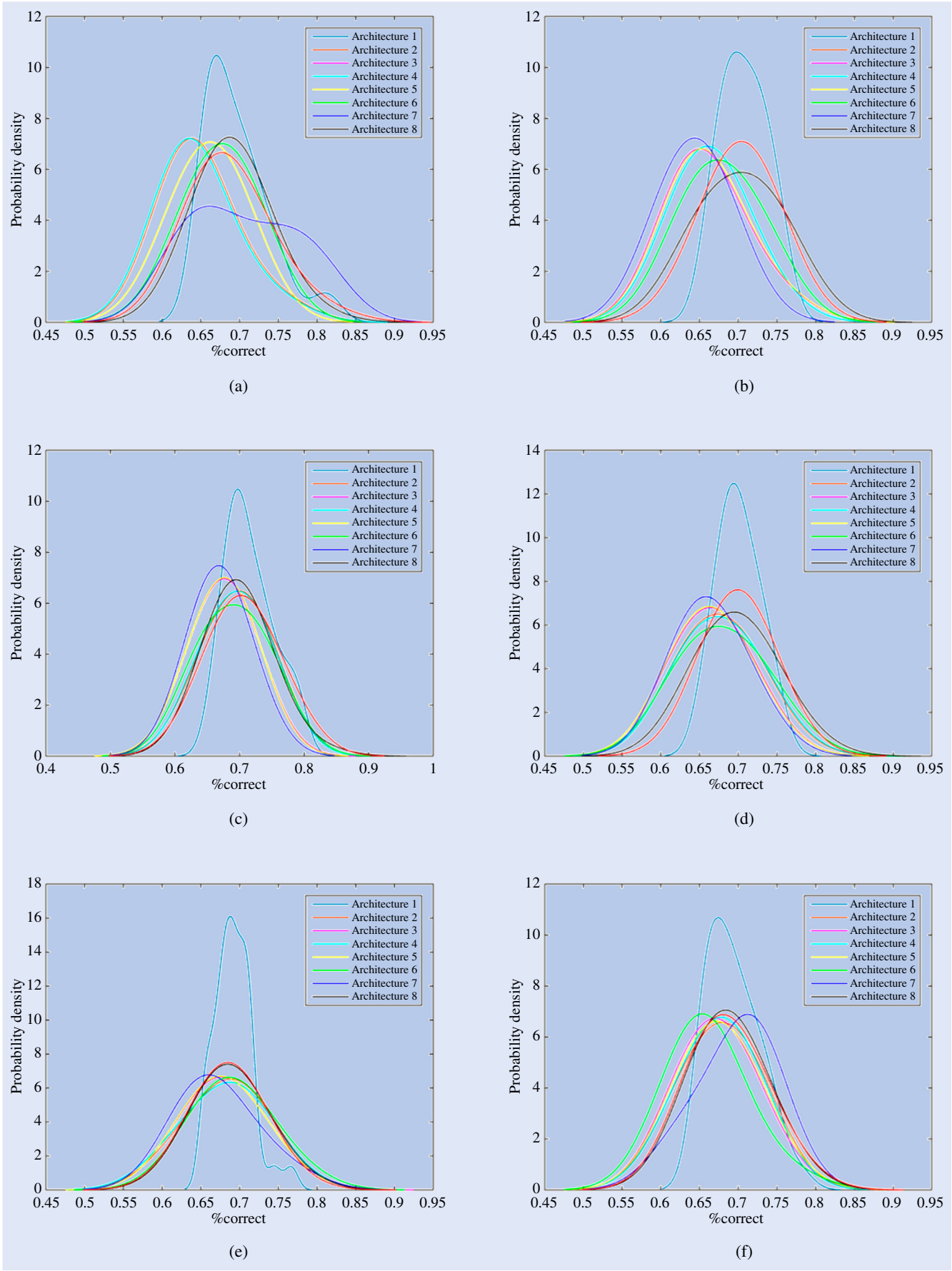

Figure A3. Clockwise, distribution of the overall correctly classified obligors for the German data. Results are based on the 0%, 50%, 60%, 70%, 80% and 90% percentile of the correlation to the target variable. (a) 0% percentile. (b) 50% percentile. (c) 60% percentile. (d) 70% percentile. (e) 80% percentile and (f) 90% percentile.